Movie Recommendation Wrap up Report

Recsys 7조 유쾌한발상
이채원, 김소미, 서현덕, 백승주, 유종문

1. 프로젝트 개요

A. 개요

전처리된 MovieLens 데이터셋의 사용자 영화 시청 이력데이터를 바탕으로 사용자가 시청했던 영화와 시청할 영화 총 10개를 추천한다. 단순히 sequential 한 예측을 하는 것이 아닌 중간의 시청 이력이 비어있기 때문에 전반적인 유저에 대한 예측을 수행해야만 한다.

추가로 영화에 대한 side information으로 장르, 개봉년도, 작가, 감독의 정보가 주어진다. 주어진 데이터들을 활용하여 사용자에게 10개의 영화를 추천하고 Recall@10 값으로 평가한다.

많은 모델들을 경험해보고 실제상황과 유사한 환경에서 보다 효과적인 추천 모델을 만들 수 있는 역량을 강화하는 것이 대회의 목표이다.

B. 활용 장비(도구)

- 개발환경: VsCode를 사용한 UpStage GPU와 SSH를 연결
- 협업툴: Notion, GitHub, MIflow
- 의사소통: 카카오톡, Zoom, Slack, 필요시에 오프라인으로 프로젝트 진행

C. 기대 효과

- 사용자에게 개인화된 영화추천 가능
- 유실된 로그 데이터 복구
- Sequential 데이터와 static 데이터가 동시에 주어질 경우에 효과적인 추천 모델 구현

2. 프로젝트 팀 구성 및 역할

전체	문제 정의, 계획 수립, 아이디어 제시, EDA, 논문 구현, 데이터 전처리,
이채원	모델 구현/실험(Rule-based, UBCF, IBCF, FFM, BERT4Rec), ensemble, 실험 정리, 회고
유종문	모델 구현/실험(SASRec, BERT4Rec, Rule-based, BPR), ensemble, 자체 모델 평가기준 생성
김소미	모델 구현/실험(S3Rec, DeepFM, BERT4Rec, SASRec)
서현덕	모델 구현/실험(DAE, AutoRec, BPR, FISM, UBCF, Multi-VAE, BERT4Rec), ensemble
백승주	모델 구현/실험(FFM, DeepFM, BERT4Rec, S3Rec), ensemble, mlflow 환경 설정

3. 프로젝트 수행 절차 및 방법

A. 목표 설정

① 최대한 다양한 모델과 데이터 처리 기법을 경험하는 것
② 논문을 읽고 구현된 모델을 이해할 수 있는 실력을 갖추는 것

B. 프로젝트 사전 기획

① 프로젝트 일정

Task	날짜	담당
BPR	2022년 4월 13일 → 2022년 4월 15일	유종문 David Seo
HARD VOTE	2022년 4월 13일 → 2022년 4월 14일	백승주 Chaewon Lee
4/11 중간점검 (brain storming)	2022년 4월 11일	
Rule base	2022년 4월 4일	유종문 Chaewon Lee
4/4 중간점검	2022년 4월 4일	
DL CF	2022년 4월 1일 → 2022년 4월 6일	David Seo Chaewon Lee
Deep FM 구현 및 실험	2022년 3월 29일 → 2022년 4월 11일	소미 백승주 유종문
MLflow 세팅 후 모듈화 해서 공유	2022년 3월 27일 → 2022년 4월 1일	백승주
S3Rec 모델 실험	2022년 3월 27일 → 2022년 4월 14일	Chaewon Lee 유종문 소미 백승주 David Seo
문제정의	2022년 3월 24일	소미 백승주 유종문 David Seo Chaewon Lee
BERT4Rec	2022년 4월 7일 → 2022년 4월 15일	유종문 백승주 소미 Chaewon Lee David Seo
EDA	2022년 3월 21일 → 2022년 4월 14일	유종문 소미 Chaewon Lee David Seo 백승주
Non-DL CF	2022년 3월 30일 → 2022년 4월 14일	David Seo Chaewon Lee
FFM 구현 및 실험	2022년 4월 11일 → 2022년 4월 14일	백승주 Chaewon Lee

② 역할 분담

대회 초반에는 Content-Based 모델 구현 팀과 Collaborative Filtering 모델 구현팀으로 역할 분담하여 모델 구현 및 실험을 진행하였고, 이후에는 여러 모델을 함께 구현하였다.

③ 대회기간에서 사용할 그라운드 룰 생성

(1) 기한 내에 못할 것 같으면 미리 말하기
(2) 언제든 도움 요청하기
(3) 코딩 컨벤션 정하기
(4) jupyter notebook파일 업로드 시, README와 markdown으로 설명 추가하기
(5) 한 방법을 다 같이 고민해보고 안되면 다음 방법으로 넘어가기
(6) 개선할 점과 단점을 이야기할 때 기분을 생각해서 이야기하기
(7) 업무 현황을 잘 공유하고 소통하기

④ GitHub 버전 관리 규칙

Commit 시, 다음과 같은 Header 양식을 사용하기로 하였다.

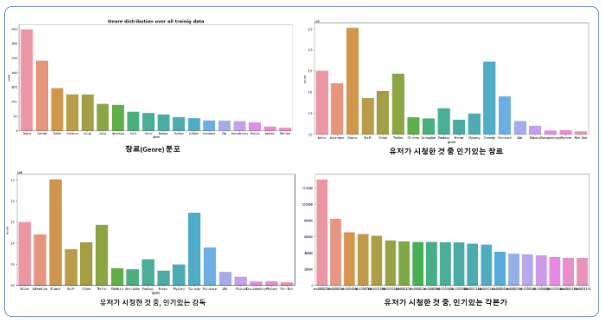
[TYPE][Author] contents (#issue number)
TYPE 종류: FEAT, FIX, DOCS, STYLE, REFACTOR, TEST, CHORE

사용할 Branch와 각 이름은 다음과 같이 정하였다.

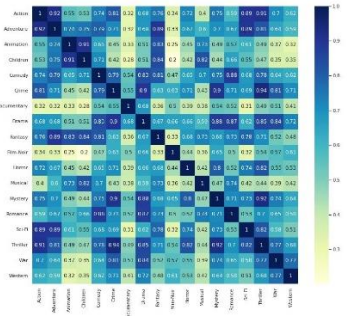
feature/{기능}, hotfix/{#issue}
사용 브랜치: main, develop, feature, hotfix

C. EDA

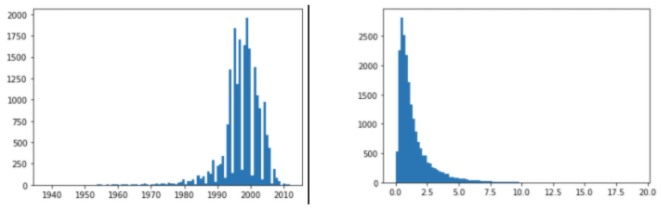
① 영화의 장르 분포, 사용자와 feature간의 상관성 분석



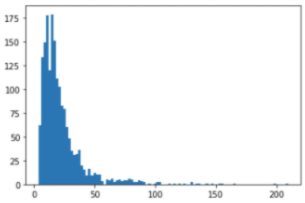
② Genre Correlation



③ 영화의 년도 별 개수와 전체 년도의 평균과 분산



④ 유저별 장르의 개수와 평균, 분산

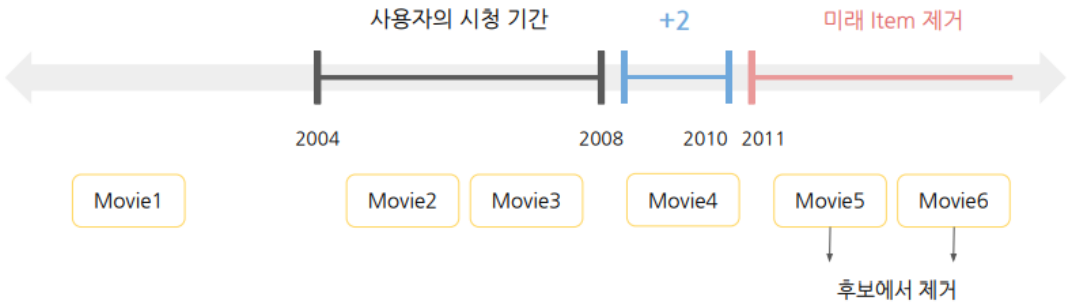


⑤ 유저별 영화 시청횟수 분포
⑥ 고전명작과 현대명작의 분포

D. Data Processing (데이터 처리)

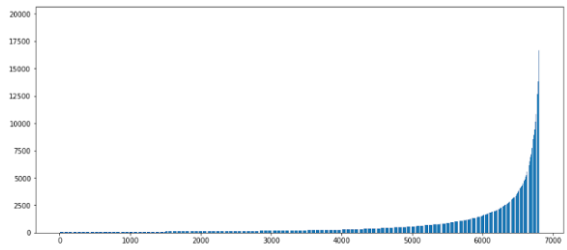
① Handling Future Item

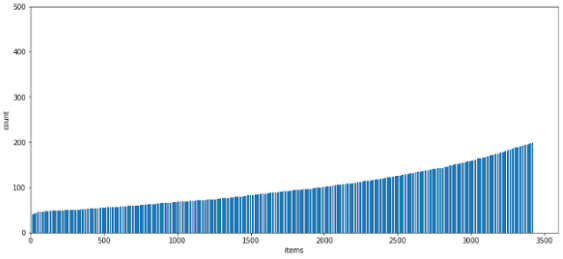
사용자는 마지막 활동기간 (last timestamp) 이후에 개봉된 영화는 보지 못하기 때문에, 최종 활동년도 + 2 이후에 개봉된 영화는 추천대상에서 제외하도록 설정하였다.



② Handling Unpopular Item

영화 중 상영횟수가 200회 이하로 나온 영화가 전체 영화 중 절반 정도를 차지하여, 모델별로 횟수를 다르게 설정하여 추천하고자 하는 item의 대상을 줄였다.





E. 모델 개요

Content Based Model	Deep FM, FFM
Collaborative Filtering	UBCF, IBCF, SVD, Multi-VAE, DAE, RecVAE, BPR, AutoRec, User Profiling
Sequential	SASRec, S3Rec, BERT4Rec
Rule Based Recommendation	Popular, Rec by genre preference, Rec by user timestamp

F. 모델 선정 및 분석 (최종 앙상블에 사용한 4가지 모델)

① BERT4Rec: 유저의 시청기록을 masking하는 cloze task로 학습하는 것이 현재 대회에서 해결하고자 하는 문제와 유사하다고 판단하여 선정하게 되었다.

② UBCF: Log 데이터가 일정하지 않기 때문에 가장 단순하면서도 성능이 좋게 나와서 baseline으로서 구현하였다.

③ Multi-VAE: VAE의 샘플링 기법을 활용하여 보지 않은 영화에 대해 더 정확한 추천을 하기 위해 선정하였다.

④ DAE: Noise를 추가하여 학습 데이터에 과적합 되는 것을 방지하기 때문에 선정하게 되었다.

G. 협업 과정

① Notion의 활용

전반적인 프로젝트 일정과 실험결과 공유, 참고자료, 질문사항 등 모든 공유할 내용들을 적었다. Dailly Checklist를 활용해서 각자가 할 일을 점검했다

② Git의 활용

GitHub의 branch를 적극적으로 활용해서 각 모델간의 실험을 독립적으로 설정하였고, 모든 팀원들이 실행한 코드들을 github를 통해 공유하며 협업을 할 수 있었다. 필요한 경우, pull request, merge conflict를 하여 더 빠른 코드 교환이 이루어질 수 있게 되었다.

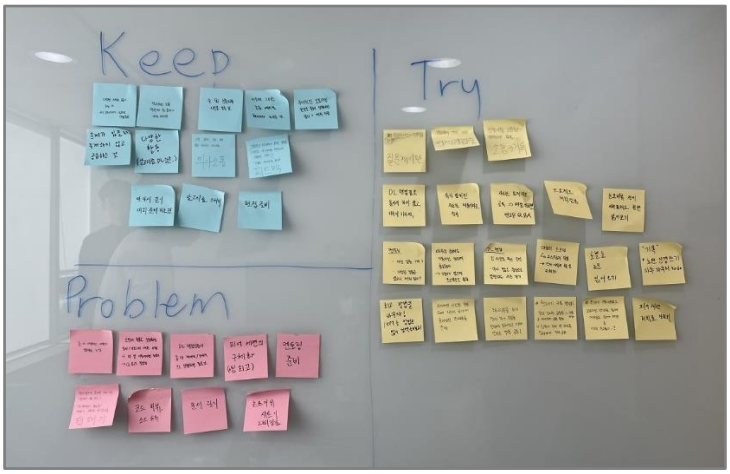
③ 실험관리를 위한 mlflow 설정

GCP 서버에 mlflow tracking server를 올려서 실험을 관리하였다. 실험에 사용된 hyper-parameter, log, artifact, model 등을 기록해서 팀원들과 공유할 수 있도록 환경을 구성하였다. (http://34.105.0.176:5000/)

							Metrics			Parameters		
Start Time	Duration	Run Name	User	Source	Version	Models	Train/loss_avera	Valid/accuracy	Valid/loss	data_split	dropout_rate	epochs
4 days ago	2.0h	drop_rate 0.5	root	train.py	16fea3	-	5.703	0.128	5.76	leave_one_out	0.5	200
5 days ago	3.9h	-	root	train.py	-	-	4.621	0.206	4.567	-	-	-
5 days ago		-	root	train.py	-	-	6.767	0.074	7.014	-	-	-
5 days ago		-	root	train.py	-	-	5.351	0.153	5.279	-	-	-
5 days ago	1.5h	-	root	train.py	-	-	5.095	0.17	4.976	-	-	-
6 days ago	1.6h	seqlen_100	root	train.py	aeee48	-	5.127	0.163	5.064	leave_one_out	0.1	200
6 days ago	54.4min	leave_one_out	root	train.py	0cf1f4	-	5.297	0.15	5.297	leave_one_out	0.1	200
6 days ago	13.0min	head_8, laye...	root	train.py	0cf1f4	-	5.773	0.126	5.792	split_by_user	0.1	200
6 days ago	3.4min	head_8, laye...	root	train.py	0cf1f4	-	6.471	0.1	6.368	split_by_user	0.1	200
6 days ago	1.7min	head_8, laye...	root	train.py	0cf1f4	-	7.735	0.003	7.744	split_by_user	0.1	200
6 days ago	14.7min	max_prob 0.3	root	train.py	0cf1f4	-	5.466	0.132	5.713	split_by_user	0.1	200
6 days ago	28.6min	seqlen_100	root	train.py	6feaad	-	5.293	0.145	5.417	split_by_user	0.1	200
6 days ago	12.7min	batch_size512	root	train.py	6feaad	-	5.51	0.133	5.704	split_by_user	0.1	200
6 days ago	18.6min	experiment	root	train.py	6feaad	-	5.485	0.137	5.67	split_by_user	0.1	150
6 days ago	5.1min	-	root	train.py	6feaad	-	7.709	0.004	7.715	split_by_user	0.1	150
6 days ago	5.1min	-	root	train.py	6feaad	-	7.692	0.004	7.704	split_by_user	0.1	150

④ KPT 회고

중간 회고를 위해 팀원들과의 오프라인 만남을 통해 Keep, Problem, Try할 것들에 대해 리스트업을 하고, 해결책을 제시하였다.



4. 프로젝트 수행 결과

A. 모델 평가 및 개선방향

① Content-Based 계열 모델 (Deep FM, FFM)

MovieLens 20M 데이터셋에서는 Contents-base 모델이 성능이 좋지 않았다. 영화에 대한 side information을 최대한 train하는것이 좋은 추천 성능을 낼 수 있을 것이라 생각했지만 많은 feature를 사용할 수록 실험 결과가 좋지 않았다. 이를 통해 데이터셋 마다 특징이 존재하고 적합한 모델을 선택하는 것이 필요하다는 것을 알게 되었다.

② Collaborative Filtering 계열 모델

구현이 빠르지만 성능이 뒤쳐지지 않아서 테스트해 보기에 좋았다. Data processing의 효과를 측정할 때도 활용되었다. 안정적이게 대부분의 모델에서 좋은 성능을 보였다. 더 많은 hyper parameter를 적용해보지 못한 것이 아쉬웠다.

③ Sequential 계열 모델

데이터 셋의 여러가지 attribute를 모두 활용하여 학습하기에 시간이 너무 오래 걸렸다. BERT4Rec 같은 경우 Item의 Sequence만 고려하고 다른 attribute를 활용하지 못했기 때문에 모델 성능이 좋지 않았다. 하지만, 앞으로 볼 영화만 추천을 하는 task에서는 좋은 성능을 보일 것이라고 보인다.

④ Rule-Based Recommendation

각 유저가 주로 시청하는 장르를 주로 추천하는 Rule-base 모델은 무엇보다 학습시간이 따로 없고 계산시간이 빠르다는 장점이 있었다. 하지만 전체의 유저에 대해서 일반화하기에는 부족했다고 판단되었다. 유저가 활동했던 년도(year)를 기반으로 인기있는 아이템을 추천한 것은 실제 유저들이 영화를 보는 모습을 잘 반영하는 것이었지만, 위와 마찬가지로 모든 유저에 대해서 일반화하긴 어려웠다.

B. 시연결과

Model	Augmentation / Skills	Recall@10
DeepFM	Genre, Writer, Director Concatenation	0.079
FFM	Genre, Writer MAE loss 적용	-
UBCF	Cosine similarity, voting, future item 제거	0.1161
S3Rec	Genre	0.0892
BERT4Rec	Top10 per 5 inference 방법 적용	0.1151
Multi-VAE	Epoch(200), WD(0.01)	0.1421
DAE	Epoch(150), WD(0.01)	0.1420
Rule by Genre	Top5 장르에 대해서 3:2:2:2:1 비율 적용	0.07
RecVAE	Epoch(50), gamma(0.004)	0.1243

C. 앙상블 (Ensemble)

독립적으로 실험한 모델들에 대해 앙상블을 진행하여 성능을 최대한으로 끌어올리고자 하였다. 여기서 사용한 앙상블 기법은 다음 2가지가 있다.

① Hard Voting: 각 모델에서 뽑은 추천 리스트에서 많이 등장한 영화 10개를 vote

② Weighted Hard Voting: 성능이 높은 모델이 추천한 영화에 가중치를 부여해서 많은 점수를 얻은 영화 10개 vote

Combination	Method	Recall@10
Multi-VAE & SASRec	Hard Voting	0.1274
DAE & UBCF	Hard Voting	0.1365
DAE & Multi-VAE	Hard Voting	0.1470
*Top 5	Hard Voting	0.1418
*Each Model	Hard Voting	0.1493
*Top 10	Hard Voting	0.1482
*Best7	Weighted Hard Voting	0.1643
*Best3	Weighted Hard Voting	0.1644
*Best4	Weighted Hard Voting	0.1675

*Best3: 가장 성능이 잘 나왔던 모델 3개 (Bert4Rec, DAE, UBCF)
*Best4: 가장 성능이 잘 나왔던 모델 4개 (Bert4Rec, Multi-VAE, DAE, UBCF)
*Best7: 가장 성능이 잘 나왔던 모델 7개 (Bert4Rec, Multi-VAE, DAE, UBCF, FFM, DeepFM, Rule Base)
*Top 10: 제출 성능이 가장 높았던 submission 10개
*Top 5: 제출 성능이 가장 높았던 submission 5개
*Each Model: 비교적 성능이 좋았던 모델들을 사용해 겹치지 않도록 모델들을 선택(Bert4Rec, S3&SASRec, DAE&MVAE, UBCF&SASRec, RecVAE, FISM&SASRec)

D. 프로젝트 결과

5. 자체 평가 의견

A. 좋았던 점

① 미련 없이 최선을 다한 점

② 근거를 기반해서 모델을 선정한 점

③ 대회기간이 길었음에도 끝까지 몰입해서 좋은 결과를 이루었던 것

④ 다양한 아이디어(Data processing, ensemble, EDA)를 생각해내고 구현한 것

⑤ GitHub의 Branch관리 및 merge conflict등 사용을 제대로 해보았던 것

B. 아쉬웠던 점

① 더 많은 EDA를 해보지 못했던 것

② 모델 탐색을 더 하지 못한 것 (EASE 모델의 사용)

③ 추천 라이브러리를 사용하지 못한 것 (RecBole)

④ 문제 정의와 이해를 깊게 해볼 것

1. 나는 내 학습목표를 달성하기 위해 무엇을 어떻게 했는가?

A. 우리 팀과 나의 학습목표는 무엇이었나?

이번 대회의 목표는 할 수 있는 많은 모델을 시도해보고 구현해보는 것이었다. Recommendation 과 관련된 많은 논문을 서로 스터디하고 구현해보며 프로그램 구현 능력을 향상시키는 것이 목표였다.

B. 개인 학습 측면

앙상블에 대한 이해: 개인적으로 저번 대회에서는 앙상블에 대해 신경을 많이 쓰지 못써서 아쉬웠다. 이번 대회에서는 팀원분이 작성한 hard-voting 하는 소스를 읽어보고 모델 성능에 따라 가중치를 줘서 hard-voting 하는 알고리즘을 만들어봤다.

Contents based 모델에 대한 이해: DeepFM, FFM 계열의 모델을 만들고 실험해 보았다. 추론 과정에서 많은 시행착오가 있었지만, 추론 데이터셋을 적절히 줄이고 추론하는 소스의 구조를 보다 효율적으로 개선하여 빠르게 만들 수 있었다. 이 과정에서 pandas, NumPy 패키지를 좀 더 잘 사용할 수 있게 되었다.

Sequential 모델에 대한 이해: Bert4rec, S3Rec 모델에 대한 논문을 팀원들과 함께 읽고 구현해보며 Sequential 모델의 구조를 보다 잘 이해할 수 있었다.

MIflow Tracking server 구축: GCP compute machine과 cloud storage를 조합하여 실험을 실행하고 관리할 수 있는 서버를 구축하였다. 이를 통해 클라우드 플랫폼을 통해 서버를 구축하는 방법을 알 수 있었다.

C. 공동 학습 측면

어려웠던 논문을 다같이 스터디 해보며 보다 쉽게 이해할 수 있었고, 함께 어려운 모델을 함께 구현해보면서 협업능력과 구현능력을 향상시킬 수 있었다.

2. 나는 어떤 방식으로 모델을 개선했는가?

A. 사용한 지식과 기술

Contents based 모델에 대한 지식을 바탕으로 DeepFM, FFM 모델 실험 소스 구현

클라우드 플랫폼 지식을 바탕으로 MIflow Tracking 서버를 구축함

프로젝트 모듈화 하는 방법을 숙지하여 model, dataset, train, inference를 다양한 hyperparameter 로 편하게 실험할 수 있도록 구현

B. 내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?

Contents based 모델은 MovieLens 20M 데이터셋에서 불리하다는 것을 확인하였다. 많은 side-information을 활용하였을 때 사용자에게 보다 좋은 추천성능을 가진 모델을 만들 수 있을 것이라 생각했지만 예상과는 다르게 side-information을 더 많이 사용할수록 모델 성능이 안 좋아지는 결과를 확인할 수 있었다.

프로젝트를 모듈화해서 구현하고 파일로 하이퍼파라미터를 관리할 수 있도록 해서 실험관리를 편하게 할 수 있었다. 보다 많은 실험을 빠르게 진행할 수 있었다.

3. 전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

각 모델을 직접 구현할 수 있는 능력을 갖추고자 노력하였고, 모델 구현, 실험, 추론할 수 있는 소스를 직접 개발할 수 있게 되었다.

MIflow를 통해 실험 환경 구축: 저번 대회에서 적극적으로 활용하지 못했던 실험 관리 도구를 이번대회에서 도입해보고자 노력했다. GCP를 통해 mlflow tracking 서버를 구축해 실험할 때 사용했던 hyperparameter, model을 저장할 수 있게 하였다. 이를 통해 팀원들과 보다 효율적으로 실험내역을 공유할 수 있게 되었다.

4. 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

직접 모든 모델을 구현해야 해서 더 많은 모델을 실험해보지 못했다. 모델에 대한 이해가 깊지 않았기 때문에 구현에 시간을 많이 쏟을 수밖에 없었고 더 많은 모델을 직접 구현해보지 못한 것이 아쉬웠다.

기본기가 약하다는 생각이 많이 들었다. Numpy, pandas를 잘 사용하지 못해 쉽게 구현할 수 있는 것을 어렵게 만들게 되어 구현 시간이 너무 오래 걸렸다는 것이 아쉬웠다.

5. 한계/교훈을 바탕으로 다음 프로젝트에서 스스로 새롭게 시도해볼 것은 무엇일까?

간편하게 쓸 수 있는 라이브러리를 활용해 실험을 더 많이 진행해보는 것, Recbole 과 같이 많은 모델들이 이미 구현된 라이브러리를 알아보지 못한 것이 아쉬웠다. 다음 대회에서는 이런 라이브러리를 적극 활용해보고 싶다.

데이터셋에 적합한 모델을 많이 찾아보지 못했다. 대회가 종료되고 Paper with Code를 확인해보니 VAE 계열 모델이 MovieLens 20M 데이터셋에서는 성능이 좋다는 것을 확인할 수 있었다. 다음 대회에서는 데이터셋에 대한 조사를 먼저 진행하여 적합한 모델을 찾는 작업을 우선적으로 수행해야할 것 같다.

1. **나는 내(팀) 학습목표를 달성하기 위해 무엇을 어떻게 했는가?**
   나와 우리 팀의 학습 목표는 모두가 추후 프로젝트를 위해 최대한 다양한 경험을 하는 것이었다. 이를 위해 개인 측면에서는 기본 강의 내용에 충실하되 필요 시 새로운 내용을 적극적으로 찾아보고자 했고, 팀 측면에서는 자율적인 의견 제시를 장려했으며 원활한 소통을 위해 다양한 협업 툴을 활용했다.

2. **나는 어떤 방식으로 모델을 개선했는가?**

   A. Model Customization
   User profiling model과 BERT4Rec inference를 customizing했다. 먼저 유저들을 마다 프로파일 벡터를 만들어서, 가장 유사한 아이템 벡터를 갖고 있는 영화 10편을 추천했다. 영화 벡터는 시청한 유저들이 비슷한 영화들은 서로 유사한 영화일 것이라는 가정을 하여, 해당 영화를 시청한 유저들을 담고 있는 원핫 인코딩 벡터로 정의했다. 유저 프로파일 벡터는 유저가 시청한 영화 벡터의 평균으로 정의했다. 각 유저 프로파일 벡터마다 코사인 유사도가 가장 높은 영화를 10편 추천했다. 또한 BERT4Rec inference는 마지막 timestamp에서 10개를 추천하는 것이 아니라, 중간에 있는 여러 개의 timestamp에서 추천하도록 한 뒤 나온 아이템의 개수로 hard voting하도록 customizing했다.

   B. 과적합 방지를 위한 모델 구현
   기본 auto encoder는 입력 데이터에 과적합 되어 좋은 성능을 내지 못했다. 따라서 입력 데이터에 noise를 추가한 후, 원래 데이터를 맞추도록 학습하는 denoising auto encoder와 학습된 평균과 분산을 통해 샘플링하는 Multi Variational Auto Encoder 모델을 구현했다. Noise 추가와 sampling을 통해 효과적으로 과적합을 해소했다.

   C. EDA & Data Screening
   유저별, 아이템별 특징을 분석했다. 유저들이 시청하는 영화들의 평균 시청 횟수, 편차, 인기 영화들의 장르 분포, 등을 분석하여 유저의 시청 성향을 파악하고, 이를 기반으로 새로운 가설들을 세우고 검증했다.

3. **내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?**
   충분히 타당한 논리가 뒷받침 될 때 성능이 높아지는 것, 상황에 맞게 모델을 customize하는 것이 효과적이라는 것, 그리고 EDA의 중요성을 깨달았다. User profiling model은 내가 주어진 상황에 맞춰 가설을 세우고 배운 것들을 종합하여 만든 custom 모델이다. 해당 모델은 당시 작업하고 있었던 SOTA 모델의 성능(recall@10=0.0892)에 준하는 성능(0.086)을 냈다. 또한 입력 데이터에 과적합 되는 auto encoder의 안좋은 성능(recall@10=0.002)을 보고 noise를 추가한 denoising auto encoder를 활용하여 0.1395라는 당시 최고 성능을 낼 수 있었다. 두 경우 모두 합리적인 근거가 있었기 때문에 성능을 높일 수 있었다. BERT4Rec은 sequential 모델로 주어진 시청 내역을 보고 다음으로 시청할 영화를 추천하는 모델이다. 하지만, 주어진 시청 내역을 랜덤하게 가리고 맞추는 방식으로 맥락을 학습한다는 점에서 이번 대회 과제와 굉장히 유사하다고 생각했다. 따라서 맨 마지막 timestamp에서 10개의 영화를 추천하는 것이 아니라, 5개의 timestamp마다 10개의 추천 영화를 추출한 뒤 등장 횟수가 가장 많은 영화를 10개 추려서 추천했다. 결과적으로 BERT4Rec은 최종 성능이 가장 좋았던 앙상블 모델(recall@10=0.1675)에서 가장 큰 비중(voting weight)을 차지했다. 모델을 우리의 상황에 맞춰 변형하는 것이 큰 효과가 있다는 것을 경험했다. 또한 나는 모델 학습에 방해가 되는 유저와 아이템을 찾기 위해 EDA를 진행했다. 유저 별, 아이템 별 특징을 분석하면서 인사이트를 얻을 수 있었고, 이는 모델 개선을 위한 발상의 전환에 도움이 됐다.

4. **전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?**
   다양한 모델을 구현했고, 데이터 분석을 통해 모델의 성능을 끌어올리고자 했다. 첫 대회 때 어려워 보여서 구현하지 않은 모델들에 대한 아쉬움이 컸기 때문에, 이번에는 스스로에게 제한을 두지 않고 구현을 시도해보기로 했다. 결과적으로 2개의 non-DL 모델, 3 DL 모델, 그리고 custom 모델 1개를 구현했다. 과정이 어려웠지만, 어려웠던 만큼 코드를 읽고, 수정하고, 작성하는 능력이 많이 늘었고 구현한 모델을 확실하게 이해할 수 있었다. 또한 데이터를 분석해서 찾은 인사이트를 모델 개선을 위해 활용하고자 노력했다. 절대적으로 성능이 높은 모델을 찾는 것은 불가능 하다는 것을 느껴, 상위권 성능을 위해서는 사용하는 데이터에 대한 공부가 필요하다는 것을 깨달았다. 따라서 유저와 아이템에 대한 심층적인 EDA를 했고, 유저와 아이템 데이터에 대해 알아낸 인사이트를 바탕으로 팀원들과 함께 성능 개선을 위한 색다른 아이디어를 떠올릴 수 있었다.

5. **마주한 한계는 무엇이며, 다음 프로젝트에서 스스로 새롭게 시도해볼 것은 무엇일까?**
   다음 프로젝트에서는 더 높은 수준의 EDA와 효율적인 구현을 시도해보고 싶다. 이번에 뒤늦게 데이터 분석을 시작해서 EDA를 원하는 만큼 하지 못했다. EDA는 잘 한다면 모든 모델에 적용할 수 있는 소중한 정보를 얻을 수 있고, 틀에 갇히지 않은 새로운 아이디어를 구상할 수 있다. 따라서 다음엔 EDA에 좀 더 집중해보고 싶다. 또한 단순히 구현하는 것에 멈추지 않고, 더 나아가 효율적인 코드 구현을 연구하고 시도해보고 싶다. 이번 프로젝트 때 동일한 내용을 구현해도 어떤 코드는 13시간, 어떤 코드는 5시간, 어떤 코드는 1분 걸리는 것을 깨달았다. 추천 시스템 현업에서는 inference time이 굉장히 중요한 요소이기 때문에, 앞으로는 코드를 하나 작성하더라도 효율성을 고려하여 빠르고 정확한 코드를 만들고 싶다.

1. **팀과 나의 학습 목표**

대회를 시작하면서 팀원 모두가 함께 이번 대회의 목표는 "학습했던 수많은 모델들을 최대한 많이 적용시켜보자" 라고 설정하였다. 그에 따라 어떠한 결과가 나오더라도 많은 모델을 직접 구현하면서 이해하고자 많은 노력을 쏟아부었다. 나도 개인적으로 단순히 설명만 듣는 것보다 코드를 통해서 많은 오류를 겪는 것이 학습하는데 더 큰 도움이 됐었기에 좋은 목표설정이라고 생각했다. 개인적인 목표로는 학습했던 모델들에 대한 논문을 많이 찾아보고 읽는 것이었다. 운 좋게 대회 중 논문 스터디도 함께 팀원들과 진행할 수 있어서 이해가 안 갔던 부분들에 대해서 모두가 함께 논의하면서 공부할 수 있었다.

2. **모델을 개선시킨 방법**

사실 이번대회에서는 하나의 모델을 개선시키기 보다는 성능을 잘 내는 모델을 찾는 것이 성능점수를 올리는데 효과가 더 컸다. 그래도 모델 하나하나에 대해서 데이터를 전처리하여 성능을 높이고자 했다. 영화를 많이 보았던 사람과 많이 보지 않은 사람으로 나누어 보기도 했고, 장르의 스펙트럼이 넓은 사람을 표준편차나 분산을 사용해서 분류하기도 했다. 이후에는 모델이 잘 예측하지 못하는 유저를 2개의 그룹으로 나누어서 각 그룹에 맞는 모델을 찾는 과정도 거쳤다.

모든 모델에 대해서 실험을 진행하고 싶었지만, 각 모델의 전처리, 입력, 출력, 추론 과정까지 모두 구현하는 것은 힘들었다. 무엇보다 각 모델의 출력이 다 다르기 때문에 추론과정을 통합시키지 못했던 것이 많이 아쉬웠다. 따라서 여기에 시간을 많이 쏟아붓게 되었다. 결과적으로 모든 모델에 대해서 실험하지는 못하고 규칙기반 모델을 포함해서 약 15개의 모델에 대해서 실험을 진행했다.

이렇게 여러 모델들에 실험을 진행했었는데도, 정말 뛰어난 성능을 보이는 모델은 없었다. 그렇게 앙상블 기법을 사용해서 개선시키려고 하였다. 단순히 여러 모델들의 결과를 종합해서 가장 많이 추천을 받은 영화 10개를 추천하는 Hard voting 기반의 앙상블로 시작해서 이후에는 각 모델의 성능에 가중치를 주어서 hard voting을 하는 과정을 진행했다. 팀 내에서는 "weighted hard voting" 이라고 불렀다. 이렇게 하니 성능개선이 크게되어서 하위권이었던 순위가 단번에 크게 상승할 수 있었다. (16→3)

3. **내가 한 행동의 결과와 깨달음**

나는 이번대회에서 주로 Sequential Model과 Rule Based 추천에 집중을 하였다. 먼저 Sequential Model중 BERT4Rec을 도입하는데 많은 힘을 썼다. 논문을 공부하던 중 해당 모델의 학습방법이 현재 우리의 대회와 너무 유사하다는 생각이 많이 들었다. 또한 Rule Based 추천은 각자가 이해하기 너무 직관적이었고, 그 근거가 타당했기에 좋은 성능이 나올 것이라고 자신있었다. 실제 우리가 어떤 영상서비스를 사용한다고 생각하고 과연 각자 어떤 영화를 볼 것인지 생각해보기도 했다. 그래서 사용자별로 선호하는 장르 top5개를 뽑아서 추천하였고, 여기에 성능을 더 높이기 위해서 사용자가 해당 서비스를 사용했을 때 보다 많이 늦게 나온 영화들은 추천에서 제거하였다(미래 영상 제거 효과). 하지만 많은 노력을 했음에도 성능이 좋게 나오지는 않았다.

분명 각각의 모델이 잘 맞추는 유저가 있을 것이라고 가정하고 앙상블을 진행하기로 하였다. 그 결과, 각자 따로 있을 때는 성능을 제대로 내지 못했던 모델들이 모이자 좋은 성능을 보여주었다. 이때 앙상블의 중요성을 다시 한 번 느꼈으며 앙상블의 기법에 따라서도 그 결과의 차이가 크다는 것을 알았다.

4. **한계와 아쉬운 부분**

이번 대회를 진행하면서 개인적으로 번아웃 현상이 많이 왔다고 느껴졌다. 내가 한 것에 비해 성과가 전혀 없자, 이것을 하더라도 좋지 못한 결과를 가져올것이라는 부정적인 생각을 하기도 했다. 하지만 열심히 하는 다른 팀원들을 보며 금방 다시 힘을 내서 다방면으로 시도를 계속하였다. 초반에 분명 목표가 "성능과 관계없이 많은 모델을 경험하자" 였지만, 정작 성능이 잘 나오지 않아서 하위권에 팀 순위가 위치하자 힘을 많이 못냈던 것도 사실이다. 이런 부분들이 큰 아쉬움으로 다가왔다. 좀 더 집중하면 최종적으로 더 좋은 결과를 얻거나, 내가 얻을 수 있는 것들이 더 많았을 것이라고 생각된다.

2번째 아쉬운 점으로는 외부 라이브러리의 사용이다. 어떻게 보면 개인 학습능력과 성능간의 Trade-Off라고 할 수 있을 것 같다. 외부 라이브러리를 사용하면 모델 구현이 간편하고 그 정확도도 이미 증명되어 있어서 빠르게 많은 실험을 진행하고 성능을 높일 수 있었을 것이다. 하지만, 우리는 그런 모델을 사용하지 않고, 직접 하나씩 구현해서 모델들을 구현하였다. 그런 과정에서 많은 오류들을 겪어 시간을 많이 지체하게 되었다. 그렇지만, 이런 부분에서 분명 개개인이 모델에 대한 이해는 훨씬 컸을 것이라고 장담한다.

마지막 아쉬운 점은 팀내에서 다시 2팀으로 나누어 다른 모델링을 진행한 점이다. 각 팀간의 정보공유가 원활하지 않아서 한 쪽 팀이 말하고 있을 때, 다른 팀은 큰 도움을 주지 못했던 짧은 시기가 있었다. 이는 확실히 팀전체가 동일한 것을 공유하고 있어야 시너지가 크게 난다고 판단을 하고 곧바로 다음 모델링부터는 전체가 하나의 모델링을 진행하자고 제안하였다. 다행히 이후부터는 팀 내의 정보공유 또는 도움의 결여 같은 문제가 발생하지는 않았다.

5. **개선사항, 새롭게 시도해볼 수 있는 것들**

다른 팀들의 솔루션을 듣고 나니 좋은 성능을 냈던 다른 모델들을 도입해서 시도해보고 싶은 생각이 많이 들었다. 또한, 스케줄링에 있어서 외부 프로그램인 WBS를 사용했으면 더 철저한 일정관리가 되었을 것이라고 생각이 된다. 이런 부분을 꼭 개선시켜서 다음 대회에 적용해보고 싶다.

이번 대회에서는 mlflow를 개인적으로 크게 활용하지 못했는데, 다음에는 꼭 mlflow의 사용법을 잘 익혀서 실험관리를 더 철저하게 할 수 있도록 만들어야겠다. 무엇보다 팀원내의 협업을 위해서 효과적인 의사소통하는 방법을 기르고 싶다는 생각이 들었다.

1. 학습목표

개인 학습 목표는 모델 구현 능력을 키우고, 학습목표에 맞춰 모델을 고도화해보며 여러가지 실험을 하는 것으로 정했다. 팀의 공동 학습 목표는 CF 부터 Sequential Model까지 여러 추천 모델을 사용해보며 최대한 많은 경험을 하고 각 모델의 장단점을 파악하는 것으로 결정했다. 개인 학습 측면에서는 Model 전체의 흐름을 이해하기 위해서 해당 모델의 논문을 읽어본 후, Code를 하나하나 뜯어보며 이해하려고 했다. 또한, 대회 목적에 맞게 Customizing 할 수 있는 방법들에 대해 생각하고 제시하였다. 공동 학습 측면에서는 여러 추천 모델 논문을 함께 읽고 토론하는 과정을 통해 모델 구현능력을 향상시켰다. 또한, 프로젝트 그라운드 룰과 Git 버전 관리 규칙을 정함으로써 협업 능력을 키울 수 있었다.

2. 모델 개선 방법

학습 목표는 사용자가 앞으로 볼 영화와 과거에 봤었던 영화 10개 예측이었으므로, Inference 과정에서 주어진 데이터에 대해서 사용자의 활동기간보다 2년 뒤에 개봉한 item(영화)들을 제거한 후에 추론한다면 성능이 개선될 것이라 생각했다. 하지만 사용자의 활동기간 전에 개봉했던 고전 영화 데이터가 많았기 때문에 생각보다 많은 Item 후보가 제거되지는 않았다. 따라서, 사용자가 200회 이상 시청하지 않은 비인기 Item 후보에 대해 한 번 더 제거해주었고, 50%정도 Item 후보를 줄일 수 있었다.

두번째로 DeepFM과 S3Rec을 통해서 모델 고도화를 진행했다. 기존 DeepFM은 attribute를 Genre만 사용하였는데, 좀 더 다양한 attribute를 사용한다면 성능이 개선될 것이라 생각했고 Writer와 Director attribute를 추가하여 학습할 수 있도록 했다. 같은 방법으로 S3Rec에도 Genre뿐만아니라 Writer embedding 학습 파라미터를 추가하여 multi attribute를 학습할 수 있도록 하였다.

3. 수행 성과 및 깨달음

Inference 과정에서 negative sampling을 통해 Item 후보를 줄이는 시도는 추론 시간은 줄이고 성능은 향상시키는 것을 알 수 있었다. S3Rec은 Sequential Model임에도 불구하고 생각보다 성능이 높지 않았다. 일반적으로 미래에 볼 영화를 예측하는 태스크에는 훌륭한 성능을 보이지만, 우리의 대회목적에는 적절하지 못했던 것 같다. 게다가, CF기반 모델에 비해 학습시간이 오래 걸려 다양한 실험을 해보지 못한 것이 아쉬웠다. 기대와 달리 S3Rec과 DeepFM 모두 Multi Attribute를 사용한 모델과 단일 Attribute만을 사용한 모델의 성능차이가 미미하였다. 이번 경험을 통해 단순히 SOTA모델을 사용하는 것보다 대회 목적에 맞게 모델을 Customizing 하는 것이 중요하다는 것을 깨닫게 되었다.

4. 새롭게 시도한 변화

모델 기능을 구현할 때 최대한 모듈화가 잘 될 수 있도록 구현을 했다. 또한 Git의 branch를 적극적으로 활용하여 백업하면서 작업을 관리했는데 중간에 실수를 하더라도 다시 롤백을 할 수 있어서 효율적이었다고 느꼈다. 특히 이번 Baseline code에 사용된 S3Rec 모델의 구조가 굉장히 복잡했는데 팀원들과 오프라인 만남을 통해 논문 스터디를 진행하면서 모델을 좀 더 깊이 있게 이해할 수 있었고, 기존에 헷갈렸던 정의에 대해서도 다시 공부하면서 기초를 다질 수 있었던 시간이었다.

5. 한계 및 아쉬웠던 점

Special Mission에서 제시한 모델말고도 다양한 모델을 조사해보지 못한 것이 아쉬웠다. 최대한 많은 모델을 경험하는 것을 목표로 했기 때문에 하나의 모델을 고도화해서 여러가지 실험을 하지 못한 것 또한 아쉬움이 남았다. 이번 프로젝트에서는 상대적으로 EDA를 적게 했는데 좀 더 유의미한 인사이트를 찾기 위해서는 초반에 EDA에 시간을 좀 더 쓰더라도 데이터 분석을 철저히 할 필요성을 느꼈다.

6. 개선사항, 새롭게 시도해볼 수 있는 것

다음 프로젝트에서는 초반 EDA 과정에 많은 시간을 쏟더라도 데이터 분석에 집중해보고 싶다. 게다가, 이번에 Inference 과정에서 시간이 너무 오래 걸렸었는데 Numpy나 Pandas에 대한 이해를 깊이하고 효율적인 코드를 구현하고 싶다. 마지막으로, Recbole과 같은 외부 라이브러리를 적극 활용하여 제한없이 다양한 실험을 해보고 싶다.

1. **나는 내(팀) 학습목표를 달성하기 위해 무엇을 어떻게 했는가?**

   추천 시스템을 시작하고 첫 대회인 만큼 많은 모델들에 이 데이터를 적용해 보고 난 후, 맞는 모델로 좁혀나가기로 했다. 논문 스터디를 통해 코드와 이론의 격차를 줄였다. 강의를 자율적으로 들으며 현재 프로젝트 내용에 필요한 내용들을 적용시켰고, 효율적으로 하기 위해서 내부적으로 팀을 나누어서 진행했다. 서로의 진행 상황을 빠르게 파악하기 위해서 협업 툴을 업그레이드해서 사용했다.

2. **나는 어떤 방식으로 모델을 개선했는가?**

   추천의 경우의 수를 합리적인 이유로 최대한 줄이는 것이 좋을 것이라고 판단하여 사용자의 사용기간 이후에 개봉된 영화는 추천 결과에서 제외하도록 하는 모델을 rule based로 구현한 후에는, 모든 모델에도 전처리로써 적용하는 것이 좋을 것이라고 생각해서 전처리하여 사용했다. 또한 기록이 하루 이내로만 있는 user를 찾아내서 어떤 경향이 있는지 확인해 보았다. 기존 user 들과 sequence 길이 외에는 많은 차이가 없어서 성능 향상에 도움이 되진 않았지만, 아이디어를 내기 위한 데이터 파악을 할 수 있었다. 기본 모델에 적절한 epoch를 찾고, 과적합을 막기 위해서 Multi-VAE 코드에 early stopping을 적용했다.

   성능이 잘 나오던 Multi-VAE의 다양한 실험의 결과를 효과적으로 비교하기 위해서 MLflow를 적용하자 제안했고, 참여하였다. 모든 계열의 모델을 다뤄보고 싶어서 FFM 구현과 학습도 진행했다.

   성능을 마지막으로 끌어올리기 위해서 hard voting대신 weighted hard voting 을 제안했다. 직접 구현과 적용을 통해서 좋은 결과를 얻어낼 수 있었다. 또한, 앙상블에 사용할 모델 선정에 있어서, 단순히 기존 성능이 좋았던 모델이 아닌 여러 다른 계열마다의 모델을 하나하나 직접 선정했고, 이 또한 성능 향상으로 이어졌다.

3. **내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?**

   UBCF를 맡아서 구현했었고, 생각 없이 일단 가장 간단한 모델을 만들어보자는 생각이었는데 생각보다 높은 성능을 보여서 이 대회의 방향에 대해서 다시 생각해 보는 시간을 가졌다. 아무리 좋은 모델이라도 task가 다르면 발휘를 할 수 없다는 생각에 모델 구조와 task를 고려해서 다음에 적용할 모델들을 찾았다. 모델 만능주의로 빠지지 않아야겠다는 생각이 들었고 다음 대회에는 문제 정의와 task 분석을 조금 더 확실하게 하고 대회를 시작해야겠다고 생각했다.

4. **전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?**

   전의 프로젝트에서 아쉬웠거나 했던 실수들을 반복하고 싶지 않았고, baseline 코드의 난이도로 인해 팀에서의 환기도 필요하다고 생각되어 KPT 회고의 방법을 공부하고 정리하여 직접 진행해 보았다. 기존의 팀 회고보다 조금 더 깊은 이야기를 할 수 있었고, 그에 대한 적절한 해결책을 고민하며 발전할 수 있는 방향을 만들어나갔다.

   코드에 대한 자신감을 가졌으면 좋겠다는 피드백을 바탕으로 무엇이든 일단 시작하고 도전해보았다. 자신감을 주기 위해서 가장 쉬운 모델을 맡아서 해보고 점점 난이도를 올리며 구현했다. 결국 모든 계열의 모델들을 하나씩 경험해 볼 수 있었고 스스로의 실력 향상이 느껴졌다.

   Github의 사용에 대한 전체적인 이해와 기술을 이전 project와 특강을 통해 익혔고, 그 내용을 활용해 보고 싶어서 이번 프로젝트에 적극 활용했다. 매 commit마다 그 동안 발전된 코드의 내용들을 정리하는 시간도 되었고, 내가 그동안 무슨 일을 한 것인지 명확하게 파악되는 장점뿐만 아니라 원래 git의 목적인 코드 저장과 공유의 편리함을 확실히 체험했다.

5. **마주한 한계/교훈는 무엇이며, 다음 프로젝트에서 스스로 새롭게 시도해볼 것은 무엇일까?**

   일단 생각했던 아이디어를 구현하고 싶어서 효율성을 생각하지 않고 그냥 구현했던 코드를 사용했을 땐 2시간가량을 기다려야 했었는데, 팀원이 고쳐주신 후에는 훨씬 단시간에 학습할 수 있었다. 라이브러리 사용과 시간 복잡도를 고려해서 코드를 수정해 보아야겠다고 생각했다.

   Wide 하게 모델 탐색을 하기로 계획했다면, 하나로 깊게 파는 것이 아닌 기본적인 성능만 측정할 수 있게끔 기존에 존재하는 소스나 라이브러리들을 미리 잘 탐색해서 기회를 효율적으로 사용할 수 있도록 해야겠다.